UNIVERSITÉ DE MONTRÉAL

CENTRE DE RECHERCHE SUR LES TRANSPORTS

CASE POSTALE 6128, SUCCURSALE A, MONTRÉAL, QUÉBEC, H3C 3J7

CRT-587

THE CAPACITATED CANADIAN POSTMAN PROBLEM

by

Serge Roy
and
Jean-Marc Rousseau

Cette étude a été publiée grâce à une subvention du fonds F.C.A.R. pour la l'aide et le soutien à recherche.

Université de Montréal
Centre de recherche sur les transports - Publication #587
January 1988

Dépôt légal - 3e trimestre 1988

ABSTRACT

The capacitated Chinese postman problem (CCPP) is one of the most important arc routing problems. This article presents two algorithms adapted to the Canada Post Corporation context (CPC context): a general heuristic for merging cycles (used to group arcs together) and an algorithm to sequence the arcs of a route (Euler's tour with penalties). These two algorithms are used and tested to solve postal distribution problems in Canada.

Key words : arc routing problem, capacitated Chinese postman problem, postal distribution, Euler's tour.

RÉSUMÉ

Le problème du postier chinois avec capacité (CCPP) est un des plus importants problèmes dans la classe des problèmes avec la demande localisée sur les arcs. Nous adaptons cette formulation au contexte de la Société canadienne des postes. Cette publication présente deux algorithmes. Une heuristique générale de fusion de cycles, utilisée pour réaliser le regroupement des arcs et un algorithme pour le séquencement des arcs d'un parcours, le tour d'Euler avec pénalité. Ces deux algorithmes sont utilisés et testés pour la résolution du problème de distribution du courrier au Canada.

Mots-clés : problème sur les arcs, postier chinois, distribution du courrier, tour d'Euler.

1. INTRODUCTION

Route-making problems can be classified according to where the demand is located on the network: e.g. 1) demand located on nodes (node routing), 2) demand located on arcs (arc routing), and 3) demand located on nodes and arcs (general routing). Desrochers (1987) and Bodin (1981) have proposed a classification for these problems. Node routing problems have been frequently discussed in the literature. The Travelling Salesman Problem (TSP) is an example, in which a vehicle must visit every node of the network and return to the original node at minimum cost. If a capacity constraint is added to the vehicle and the total demand exceeds the capacity of one vehicle, a vehicle routing problem (VRP) is obtained. When the demand to be satisfied lies on the arcs of the network, the equivalent to the TSP is the Chinese postman problem (CPP). This problem consists of covering, at minimal cost, all the arcs of a network with a route starting and ending at a specified orginating point. Originally, this problem was defined on a nondirected network. Edmonds & Johnson (1973) present a polynomial algorithm for solving it. Many variants of this problem exist according to the nature of the arcs of the network: e.g. 1) completely directed network, 2) mixed network with even degree nodes, and 3) mixed network with some odd degree nodes. This third variant is very complex. Papadimitriou (1976) shows that it is "NP-hard". For a description of these variants and a solution to the third variant, see Minieka (1979).

A generalization of the CPP is introduced by Stricker (1970) and Christofides (1973). If the generation of more than one route to serve the demand is necessary because of a capacity constraint, the capacitated Chinese postman problem (CCPP) is obtained. This is a very complex problem that is "NP-hard", but has many real-life applications: e.g. 1) mail distribution, 2) road maintenance, i.e. snow removal and salt application,

cleaning, circulation line tracing, etc., 3) garbage collection, 4) rural school transportation, 5) electric meter reading, 6) parking meter collecting, 7) inspection of gas mains and electric lines, etc.

Christofides (1973) gives a heuristic and calculates a lower bound for this problem. Golden and Wong (1981) offer a mathematical formulation of the problem, correct Christofides' lower bound, and present several heuristic methods. These methods were merging cycles together or/and augmenting cycles with individual arcs. In a subsequent publication, Golden et al. (1985) present two heuristics: i) path-scanning algorithms, which construct one cycle at a time, ii) augment merge algorithms, which construct the cycles in a parallel manner. These heuristics are compared with those proposed by Christofides on several test problems. In 1985, Pearn et al. (1986) introduced a new technique to calculate a lower bound, "Node Scanning Lower Bound (NSLB)" and compared it with the "Matching Lower Bound (MLB)" proposed by Golden and Wong (1981). These authors presented particular CCPP cases, which had a special structure (network, demand, or cost), for which polynomial algorithms exist. Pearn et al. (1986) established an equivalence between demand on node and demand on arc problems by supplying a transformation of the arc problem into a node problem. The number of nodes of the resulting problem is three times the number of arcs of the original problem.

Levy (1987) and Levy and Bodin (1987, 1989) have studied a postman problem in the context of the U.S. Postal services. We will note later on the essential differences between our two problems. We will also see that the cycle construction method used by these authors can be described in part using the general framework which we propose for our approach.

We will adapt the mathematical formulation of the CCPP to the Canada Post Corporation (CPC) context. No capacity constraints on the route are present in the CPC context, but there is a time limit on each route.

Furthermore, this time limit includes the travel time between a central depot and the beginning and end of the route. Moreover, the model is generalized to consider routes with different time limits (morning routes are longer than afternoon routes). We present here a general heuristic to solve the problem under certain hypotheses. To solve the sequencing problem, we define an Euler's tour problem with penalties. We assign a penalty to each change of arc; this penalty corresponds to street crossing and the change of streets when passing from one arc to another. A straightforward heuristic method is proposed to find the Euler's tour that minimizes the sum of these penalties.

2. THE MODEL

2.1 Integer programming formulation for the CCPP

A mathematical programming formulation for the CCPP is given below. This formulation is borrowed from Golden and Wong (1981).

n : the number of nodes in the network;

k : the number of available postmen;

W : the postman's capacity;

c_{ij} : the length of an arc (i,j);

q_{ij} : the demand on an arc (i,j);

x_{ij}^{p} : $\begin{cases} 1 & \text{if postman P travels along arc } (i,j) \\ 0 & \text{otherwise} \end{cases}$;

ℓ_{ij}^{p} : $\begin{cases} 1 & \text{if postman P services arc } (i,j) \\ 0 & \text{otherwise} \end{cases}$;

E : the set of all arcs in a network.

(CCPP) Minimize $\sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{p=1}^{k} c_{ij} x_{ij}^{p}$

subject to

i) flow conservation constraints

$$\sum_{m=1}^{n} x_{mi}^{p} - \sum_{m=1}^{n} x_{im}^{p} = 0, \quad \begin{matrix} i=1,\ldots,n \\ p=1,\ldots,k \end{matrix} \qquad (1)$$

ii) all arcs must be served

$$\sum_{p=1}^{k} (\ell_{ij}^{p} + \ell_{ji}^{p}) = 1 \qquad \text{for } (i,j)\in E \qquad (2)$$

iii) all arcs served must be crossed

$$x_{ij}^{p} \geq \ell_{ij}^{p} \qquad \begin{matrix} \text{for } (i,j)\in E \\ p=1,\ldots,k \end{matrix} \qquad (3)$$

iv) capacity constraints

$$\sum_{i=1}^{n} \sum_{j=1}^{n} \ell_{ij}^{p} q_{ij} \leq W \qquad p=1,\ldots,k \qquad (4)$$

and equations prohibiting the formation of illegal subtours: (see Golden and Wong (1981) for more details)

$$x_{ij}^{p}, \ell_{ij}^{p} = 0,1 \qquad (5)$$

2.2 The Canada Post Corporation context

In the CPC context, we prefer to use the term "letter carrier" instead of "postman" because the term "postman" at the CPC is reserved for a different category of workers. The problem at the CPC consists of preparing letter carriers routes for mail distribution in the area served by a postal station. Figure 1 describes a typical letter carrier workday. A letter carrier starts and finishes his workday at the postal station. In addition, he must come back to the postal station for his lunch hour. All transportation time whether by bus or taxi is included in his workday. The

duration of the morning route must be approximately 300 minutes (including 17.2 minutes for breaks) and the duration of the afternoon route must be 180 minutes (including 15 minutes for breaks). Both these times include the transportation time to and from the postal station. Relay boxes are placed on the letter carrier's route so that he can collect along his route the mail to be distributed. Therefore, a letter carrier's route has no capacity constraint, only time limits.

In the U.S. context described by Levy and Bodin (1987), the letter carrier uses a postal service vehicle to travel from the postal station to the starting points of its delivery routes. This travel time is negligeable and is not considered. In addition, the problem is to determine delivery routes around the starting points with equal duration corresponding to the maximum load of a postal bag.

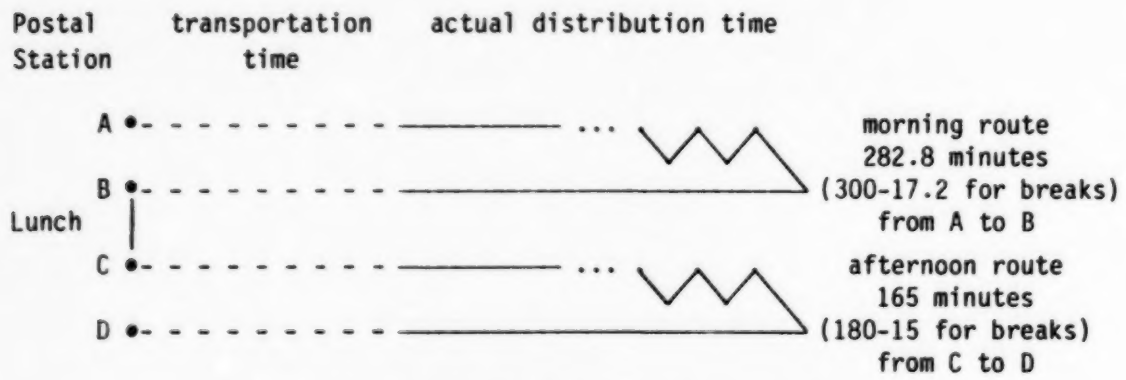


Figure 1 : A typical letter carrier workday

The following formulation is an adaptation of the CCPP formulation for the CPC context. The objective function seeks to minimize the total transportation time.

L : the number of routes (= 2·k);

W(p) : the time limit for route p;

t_{ij} : the crossing time of arc (i,j) without servicing it;

q_{ij} : the service time of arc (i,j) (not including the crossing time).

(CPC)

$$\text{Min.} \quad \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{p=1}^{L}(x_{ij}^{p} - \ell_{ij}^{p})\,t_{ij} + \overbrace{\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{p=1}^{L}\ell_{ij}^{p}(q_{ij}+t_{ij})}^{\text{constant}}$$

$$\text{s.t.} \quad \sum_{m=1}^{n} x_{mi}^{p} - \sum_{m=1}^{n} x_{im}^{p} = 0, \qquad i=1,\ldots,n;\ p=1,\ldots,L \qquad (1)$$

$$\sum_{p=1}^{L}(\ell_{ij}^{p} + \ell_{ji}^{p}) = 1 \qquad \text{for } (i,j)\in E \qquad (2)$$

$$x_{ij}^{p} \geq \ell_{ij}^{p} \qquad \text{for } (i,j)\in E;\ p=1,\ldots,L \qquad (3)$$

$$\sum_{i=1}^{n}\sum_{j=1}^{n}[(x_{ij}^{p} - \ell_{ij}^{p})\cdot t_{ij} + \ell_{ij}^{p}(t_{ij}+q_{ij})] \leq W(p) \quad p=1,\ldots,L \qquad (4)$$

Equations to prohibit the formation of illegal subtours:

$$x_{ij}^{p},\ \ell_{ij}^{p} = 0,1 \qquad (5)$$

In order to simplify the solution method, we have introduced some additional realistic hypotheses. Firstly, we assume that the graph under consideration is connected and that all arcs require services. In practice, we eliminate from the network a pair of opposing arcs between node a and b:

i) if neither of these arcs require services;

ii) if the detour imposed on the network to go from a to b is not too large (the maximum allowable detour is set at 3 times the distance between a and b). It also ensures that the network remains connected.

All remaining arcs are assumed to require services. This hypothesis has not been found restrictive in practice because in general most streets do require services on both sides and that most arcs not requiring services must be traversed to complete the Euler tour. Furthermore, some marginal

improvement method or the exact solution of the Chinese postman problem before producing the final solution could get rid of most of the unnecessary arcs remaining.

Secondly, we assume that all nodes of the graph are of even valence and that each route will start and end at the same node. These hypotheses are necessary to facilitate the solution of the Euler tour in each route generated. In practice, it is desirable to have both opposing arcs in the same route. This latter hypothesis is not very restrictive because Canada Post Corporation requests that as much as possible the letter carrier should service all arcs crossed from the starting point of a route to its end point. We suppose that with the values of L and W(p), p=1,...,L, there is a solution where all the arcs (i,j) with $x_{ij}^p > \ell_{ij}^p$ are used only for transportation.

Let us define the following:

D : the depot node in the network;

d_i : the minimum time from node D to node i and node i to node D;

$$s_i^p = \begin{cases} 1 & \text{if node i is a starting node for route p.} \\ 0 & \text{otherwise} \end{cases}$$

With the previously mentioned assumptions, we obtain the route structure described in figure 2.

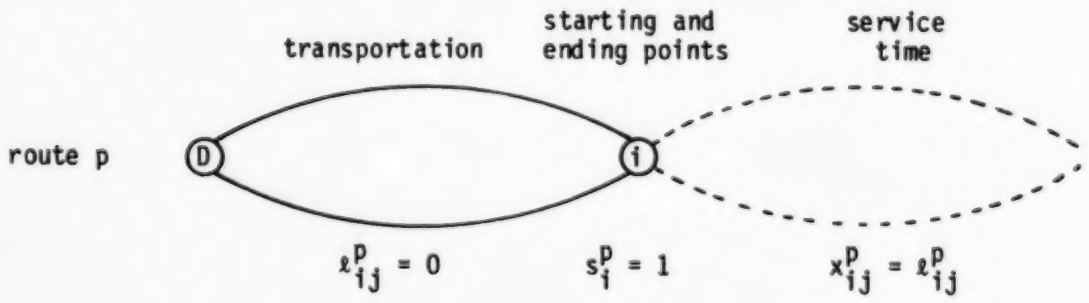


Figure 2: The route structure

Then we obtain the following mathematical programming formulation:

$$\text{Minimize} \quad \sum_{p=1}^{L} \sum_{i=1}^{n} s_i^p \, d_i$$

$$\text{subject to} \quad \sum_{m=1}^{n} \ell_{mi}^p - \sum_{m=1}^{n} \ell_{im} = 0, \qquad \begin{matrix} i=1,\ldots,n \\ p=1,\ldots,L \end{matrix} \qquad (1)$$

$$\sum_{p=1}^{L} (\ell_{ij}^p + \ell_{ji}^p) = 1 \qquad \text{for } (i,j) \in E \qquad (2)$$

$$\sum_{i=1}^{n} \sum_{j=1}^{n} \ell_{ij}^p (t_{ij} + q_{ij}) + \sum_{i=1}^{n} s_i^p \, d_i \leq W(p) \qquad p=1,\ldots,L \qquad (3)$$

$$\sum_{i=1}^{n} s_i^p = 1 \qquad p=1,\ldots,L \qquad (4)$$

$$s_i^p \leq \sum_{m=1}^{n} \ell_{im}^p \qquad i=1,\ldots,m \qquad (5)$$

$$s_i^p = 0,1$$

$$\ell_{ij}^p = 0,1$$

The problem consists of selecting L nodes from the network so that:

a) the sum of transportation times is minimized;

b) the duration of a route starting and ending at any of the nodes selected does not exceed a time limit. A minimum duration time for a route could also be added since routes with working loads that are too small are not acceptable;

c) the set of all the cycles covers the entire network.

One easily realizes the complexity of trying to solve optimally this NP-hard problem. Heuristic approaches must be used in this context.

In fact, this formulation corresponds to a location-routing problem. There is a variable cost d_i associated with the opening of a route (distribution costs) at node i. In our case, there is no cost associated with the routing aspect itself, but there are feasibility constraints. In his survey, Laporte (1988) describes two types of heuristic approaches for location-routing problems: location-allocation-routing (LAR), and allocation-routing-location (ARL).

In the LAR approach, the right number of depots is located first; clients are then assigned to depots and the routing of vehicles is performed. In the context of letter carrier problems, a heuristic method based on this approach has been proposed by Levy and Bodin (1989).

In the second approach, depots are located first at every possible location point and the corresponding routing is carried out. Sequentially, a heuristic method chooses the least attractive depot and reallocates the routing tasks to other depots until the proper number of depots is reached.

The general heuristic method we propose in the next section is based on this latter approach. We initially suppose that each node is a candidate for being the starting node of a route with a small route cycle to be serviced. Sequentially, the least interesting starting nodes are eliminated and the route cycle they service is allocated to another starting node.

3. GLOBAL PROBLEM SOLUTION PROCESS

The global problem is divided into three subproblems which are solved sequentially. The first subproblem involves the grouping of arcs to create morning and afternoon routes. To solve this problem, a general heuristic for generating cycles is used. The second subproblem is the sequencing of arcs; this generates an Euler tour which minimizes certain

penalties. The latter subproblem consists of pairing a morning route with an afternoon route in order to form a workday. To solve this subproblem, a standard assignment algorithm is used.

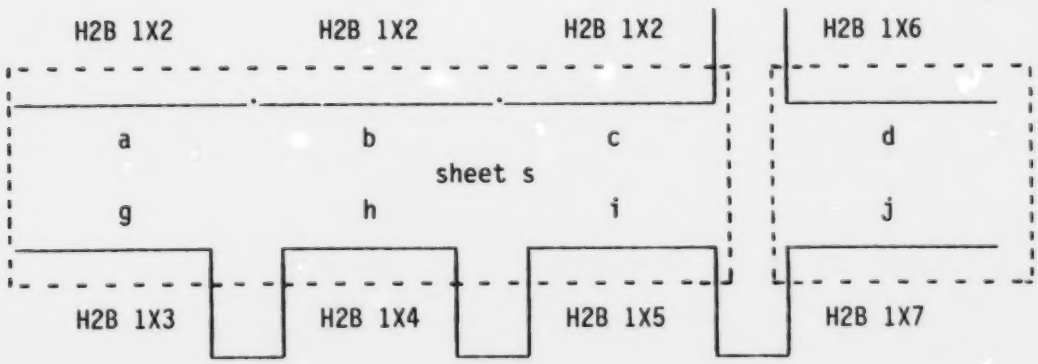
3.1 Generation of routes (the general heuristic)

The general heuristic is composed of basic operators which manipulate objects corresponding to subsets of the network. We use the same principle as in Potvin (1987) for the node routing problem. We can identify three categories of objects:

arc : a side of a street between two intersections;

basic units : a group of arcs with the same postal code, with all their opposing arcs. We assume that all arcs in the network occur in pairs of opposing arcs;

cycle : a connected group of basic units.



basic units on : - a, b, c, g, h, i;
cycles are - d, j;

Figure 3: Grouping arcs in basic units

The objects have attributes which provide information for the selection mechanism. The key attributes used are:

i) for the arc, basic units, or cycle : - transportation time, the time required to arrive at the object without servicing any arcs;
- total demand (the service time).

ii) for the cycle : - category of route (morning, afternoon, undefined);
- maximum time associated with the category (300, 180, nil).

The objects are manipulated by four operators, corresponding to distinct phases of the general heuristic. These operators are:

1) INIT : operator to initialize the basic units;
2) AUGMENT : operator to aggregate basic units into cycles;
3) FIX : operator to fix the cycle's category;
4) INSERT : operator to recursively insert free basic units into a cycle.

Each operator selects and manipulates one or more objects. We define a priori which selection functions are to be used at each step of the algorithm. A new algorithm can be created just by deciding on a new set of selection functions to be used. Changing a selection at a step of the algorihtm is as easy as changing any other user parameter in the algorithm. We can determine an object selection function by choosing:

1) an attribute optimizer (e.g. transportation time);
2) a choice optimizer (e.g. max);
3) a subset of cycles (e.g. morning cycle).

The above example would then select the morning cycle with the maximum transportation time. We will now describe the general heuristic. Fundamentally, it is composed of three steps (see Figure 4).

The first step consists of creating basic units. This is performed by the operator INIT (see Figure 5). It groups together arcs having the same postal code and all their opposing arcs (see Figure 3).

The second step of the general heuristic consists of grouping the arcs to create morning and afternoon routes. As long as categories of routes are generated, the following steps must be carried out:

1) Call operator AUGMENT which selects cycles and groups them to create routes (see Figure 6).

2) Call operator FIX which selects from the previously established cycles those to be held to create a route (see Figure 7). In the CPC context, the cycles with maximum service time are selected sequentially until the observed number of routes (a.m. or p.m.) is attained.

3) Decompose unselected cycles into their basic units, to reinitialize the problem for the next step.

Operator AUGMENT (Figure 6) requires further explanation.

In our case, AUGMENT is called with a different set of parameters for the generation of morning and afternoon routes. First a cycle C1 is selected according to criterion 1 (e.g. with the maximum transportation time). Then, a cycle C2 connected to C1 is selected according to criterion 2. If such a C2 is found, C1 and C2 are merged. If no such cycle is found, we look for the possibility of finding a basic unit E, connected to C1, that belongs to an adjacent cycle smaller than C1 and that can be transferred to C1 without disconnecting the cycle from which it comes, while respecting the capacity and other constraints of C1. If such a basic unit is found it is transferred to C1. Otherwise, a new cycle C1 is selected and the procedure is restarted. The connectivity of a cycle is checked with a simple labelling setting method.

The third and final step of the general heuristic consists of inserting into routes the basic units that were not inserted in Step 2. First, these insertions must respect all constraints of the cycles: in our case, the time limit constraint. Later on, it may be necessary to relax these constraints in order to insert all unselected basic units. This is

done by relaxing progressively the constraint by a small percentage (e.g. in our case, we relax the time limit constraint, 5 minutes by 5 minutes). The operator INSERT is described in Figure 8. It is a general exchange procedure that works recursively. At the first level, it chooses an unselected basic unit E1 and tries to insert it into a cycle C1 to which it is connected. It just tries all cycles connected to E1 for possible insertion. If this is not successful, it chooses a basic unit E2 from C1. This unit is selected large enough to allow E1 be inserted into C1 without violating the constraint if E2 is successfully inserted into another cycle. It then considers E2 as the unit to be inserted with the same process. If the insertion of E2 is successful E1 is inserted into C1. If not, it looks for another basic unit in C1 to be removed and inserted somewhere else. The procedure is not allowed to go too deep into the recursive process and is stopped after a certain number of unsuccessful trials, and restarted with a new cycle C1. This procedure is defined recursively in Figure 9 and may involve a large number of cycles. It generalizes the more simple and well-known exchange (or swapping) procedures between two cycles.

In our general framework, the cycle construction procedure used by Levy and Bodin (1987) could be described as follows. Following a seed selection procedure that we do not have in our context but that could easily be added to the general heuristic they would use service demand as the attribute in the AUGMENT procedure. They would first select cycle C1 in the subset of seeds having the minimum cumulated service demand. They will then select among all cycles connected to C1 the cycle with maximum service time to merge it with cycle C1. As for the INSERT mechanism, they use a simpler exchange procedure that is not recursive.

Steps of the general heuristic	Specific application of the general heuristic to the CPC context
STEP 1	
• Call operator INIT (see fig. 5)	
STEP 2	
N ← number of cycle categories	→ 2 (a.m. and p.m. cycles)
WHILE N > 0 DO	
• call operator AUGMENT (see fig. 6)	
• call operator FIX (see fig. 7)	
• decompose undefined cycles in basic units	
N ← N - 1	
STEP 3	
C ← insertion constraints	→ no relaxation of time constraint; tolerance ← 0
S_E ← set of free basic units	
WHILE $S_E \neq \emptyset$ DO	
E ← select free basic units	→ maximum service time
WHILE $E \neq \emptyset$ DO	
• call operator INSERT (E, C) remove E from S_E	
E ← select free basic units	→ maximum service time
S_E ← set of free basic unit	
IF $S_E \neq \emptyset$	
THEN C ← relax insertion constraints	→ tolerance ← tolerance + 5

Figure 4: The general heuristic

General operator INIT

Specific application of the operator INIT in the CPC context

```
e ← Ø
R ← set of streets
r ← select the first street

WHILE  r ≠ Ø  DO
    i ← select the first arc of the street
    WHILE  i ≠ Ø  DO
        IF [stopping criterion]                    →   not same postal code
                                                           AND
                                                       no opposing arc already
                                                         in the basic unit
        THEN
            e ← e + 1
            create a new basic unit with arc i
        ELSE  add arc i to basic unit e
        i ← select the next arc of the street
    n ← select the next street
```

Create e undefined cycles with each basic unit e.

Note: arcs have been sequenced.

Figure 5: Operator INIT - Regroup arcs into "basic units"

General operator AUGMENT	Specific application of the operator AUGMENT in the CPC context	
	Morning route	Afternoon route
S ← set of cycles		
C1 ← select an undefined cycle according to [criterion 1]	→ maximum transportation time	minimum transportation time
WHILE C1 ≠ ∅ DO		
C2 ← select a cycle connected to C1, that can feasibly be merged with C1, according to [criterion 2]	→ minimum transportation time	maximum transportation time
IF C2 ≠ ∅		
THEN Merge C1 and C2		
ELSE E ← select a basic unit connected to C1, that can feasibly be transferred to C1, according to [criterion 3]	→ minimum transportation time	maximum transportation time
IF E ≠ ∅		
THEN transfer E to cycle C1		
ELSE remove C1 from S		
C1 ← select an undefined cycle according to [criterion 1]	→ maximum transportation time	minimum transportation time

Figure 6: Operator AUGMENT - Aggregate basic units into routes

General operator FIX	Specific application of the operator FIX in the CPC context	
	Morning	Afternoon
N ← number of routes to fix	→ k	k
CA ← category to fix	AM	PM
S ← subset of undefined cycles		
C1 ← select cycle from S according to criterion 1	→ maximum service time	maximum service time
WHILE (N > 0) AND (C1 ≠ ∅) DO		
C1 ← select a cycle from S according to criterion 1	→ maximum service time	maximum service time
Fix cycle C1 into category CA		
N ← N - 1		
Remove C1 from S		

Figure 7: Operator FIX

General operator INSERT	Specific application of the operator INSERT in the CPC context
E1 ← E C1 ← select a cycle connected to E1 according to [criterion 1]	→ maximum residual time with respect to the time constraint (either A.M. or P.M. time constraint)
WHILE (C1 ≠ ∅) AND (E1 is not inserted) DO	
IF the insertion of E1 into C1 satisfies the constraint C THEN Insert E1 to C1 ELSE C1 ← Select another cycle connected to E1 according to [criterion 1]	→ maximum residual time with respect to the time constraint (either A.M. or P.M. time constraint)
(Recursive Step) C1 ← Select a cycle connected to E1 according to [criterion 1]	→ maximum residual time with respect to the time constraint (either A.M. or P.M. time constraint)
WHILE (C1 ≠ ∅) AND (E1 not inserted) DO	
E2 ← Select a basic unit from C1 (that allow the insertion of E1 into C1 if E2 is tranferred to another cycle according to [criterion 2]	→ minimum service time
WHILE (E2 ≠ 0) and (E1 not inserted) DO	
Call operator INSERT (E2, C) IF E2 inserted THEN Insert E1 into C1 ELSE E2 ← select another basic unit not already examined from C1 allow the insertion of E1 into C1 if E2 is transferred to another cycle) according to [criterion 2]	→ minimum service time
IF E1 is not inserted THEN C1 ← select another cycle connected to E1 and not already examined according to [criterion 1]	

Note: Stopping criteria are further inserted to insure that the recursive process does not go too deep (e.g. maximum 7 levels of recursions) and the total number of trials is not too large (e.g. 50 trials) for a given basic unit.

Figure 8: Operator INSERT (E,C) - Allows recursive insertion of entity E according to constraints C

3.2 Sequencing each route (Euler's tour with penalties)

Let C be a cycle (a connected and undirected subgraph with all vertices of C of even valence). An Euler tour on C means a walk along the edges of C, each edge visited exactly once returning to the starting point. There are several Euler tours for a cycle. We want to obtain the Euler tour with least road crossings and street changes in order to minimize the waiting time at intersections and the time it takes to sort the mail.

We define the penalty p(i,j) that corresponds to the execution of the sequence of arc i after arc j. These penalties have been set after discussion with Canada Post Corporation.

$$p(i,j) = \begin{cases} 22 & \text{if arc j is the other side of arc i (maximum penalty)} \\ 10 * & \text{(number of road crossings)} \end{cases}$$

and p(i,j) = p(i,j) + 1 if there is a street change.

The objective is to obtain an Euler tour which minimizes the total penalty (see Figure 9 below).

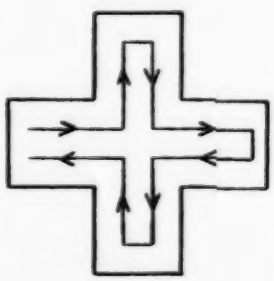


Figure 9: An Euler cycle

We can transform an Euler's tour with penalty into a TSP (travelling salesman problem). To obtain the corresponding TSP we first transform the arc routing problem into a node routing problem using the approach of Pearn (1984). We then add arcs to the new network to represent the penalty of changing arcs.

Figure 10 illustrates the new network. The broken lines represent the penalty arcs.

Following the transformation process described, given that the original network contains m arcs, we obtain a symmetric TSP with 3m+1 nodes and for each intersection of k nodes in the original networks we add $(k^2 - k)/2$ with arcs with a penalty cost.

The heuristic algorithm we used to solve the Euler tour with penalties uses the very simple nearest neighbor strategy that we apply directly to the arc problem. At each node of the network we service next the closest (according to penalty) unselected arc. This stategy may generate a tour that does not cover all arcs. A leftover arc is then selected and the procedure reinitiated. Each subtour is then merged with the closest subtour at minimum additional cost. The resulting Euler tour has proven to be very satisfactory in all cases. The method is illustrated in figure 11. Because of the very special structure of our TSP problem, we believe that the worst case analysis of this procedure would prove to be much better in this case than the worst case analysis of the nearest neighbor strategy for the general TSP.

3.3 Generation of workdays (assignment algorithm)

Consider the assignment problem in which p morning routes are to be assigned to p afternoon routes. If the morning route i is assigned to the afternoon route j, the cost incurred will be C_{ij}. We want to find an assignment with minimum total cost.

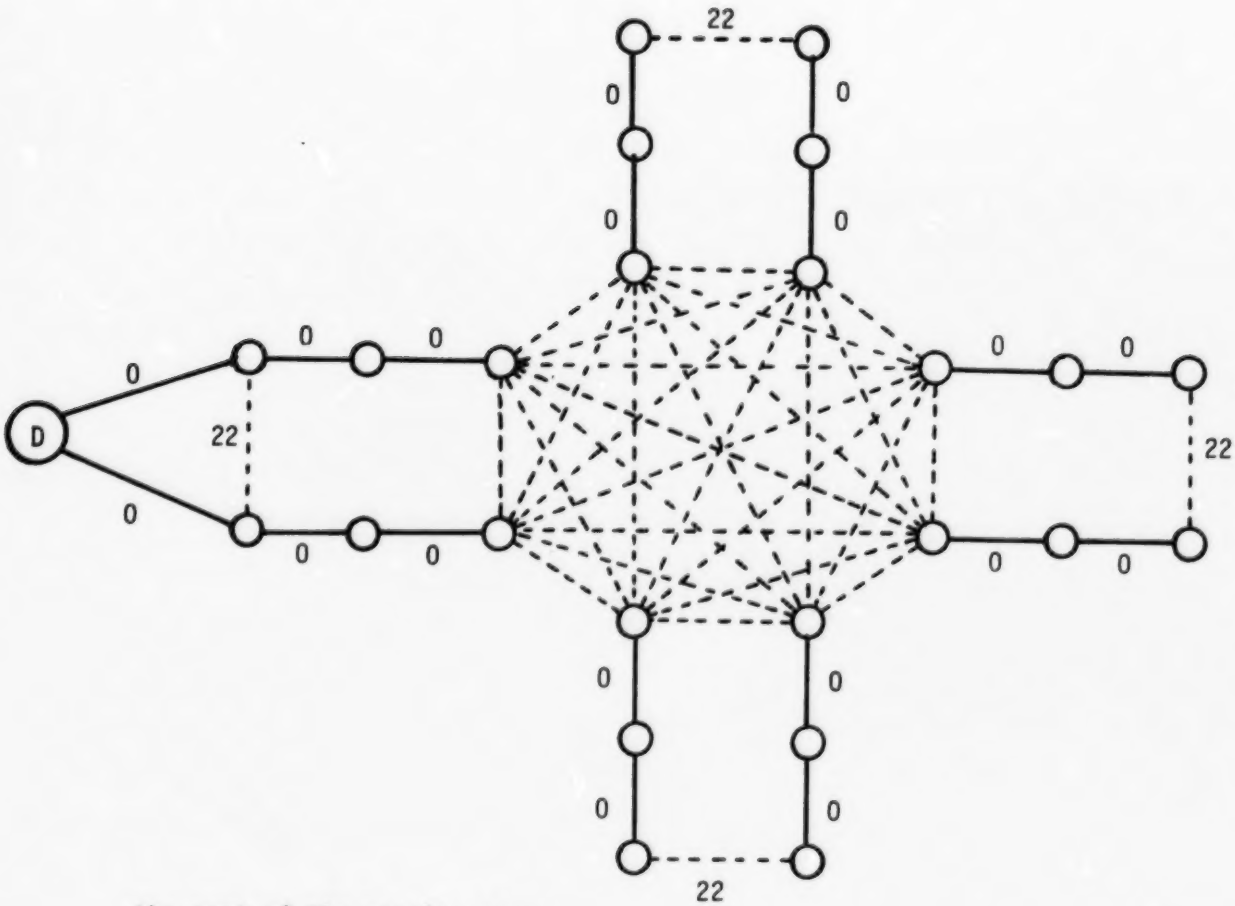
We define the following:

K : the number of available letter carriers;

M(p) : the total time of the morning route p, p=1,...,K;

A(p) : the total time of the afternoon route p, p=1,...,K;

d(i,j) : the distance from the starting node of the morning route i to the starting node of afternoon route j.



the cost of the penalty arcs

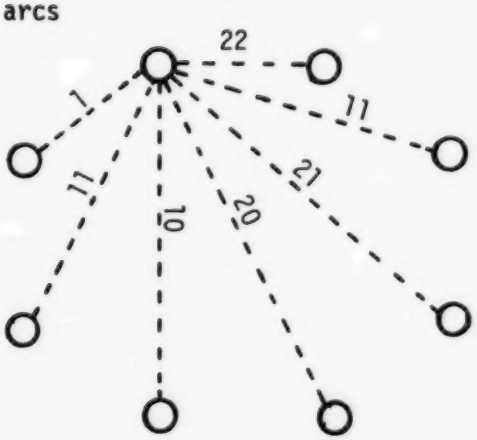


Figure 10: The TSP resulting from the transformation

Example

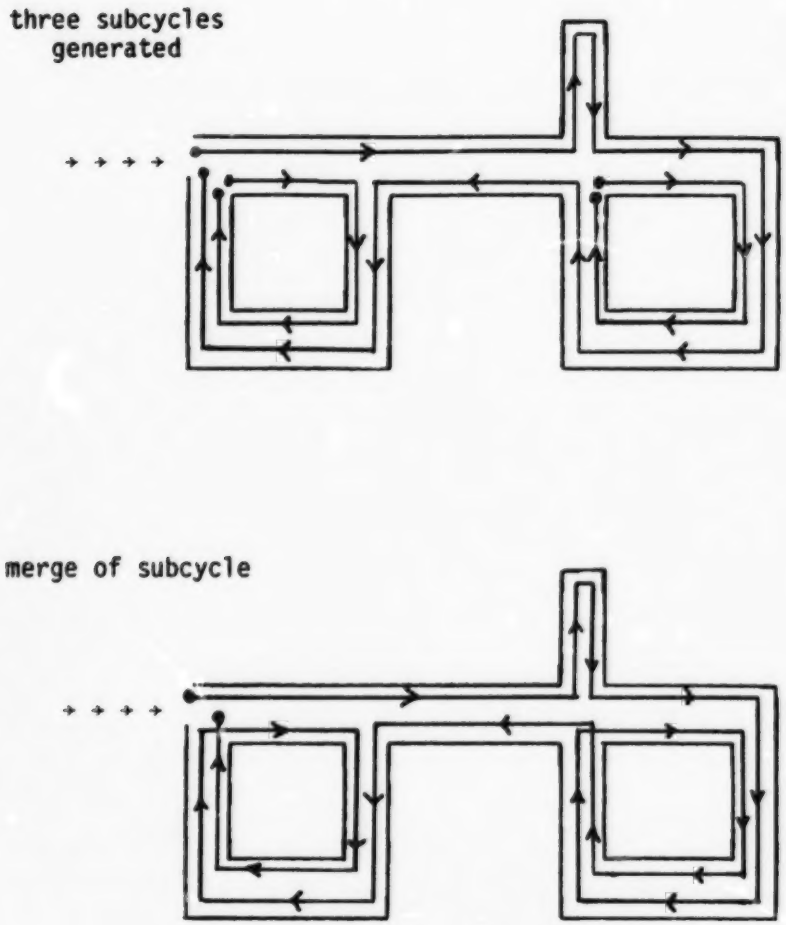


Figure 11: Solving an Euler tour with penalties

The user desires workdays to be in equilibrium (within an interval of 465 to 480 minutes), and afternoon routes to be situated within 1500 feet from the morning route. To take these constraints into account, we have defined the cost matrix C_{ij} as follows:

$$C_{ij} = [\max\{0,\ M(i) + A(j) - 480,\ 465 - M(i) - A(j)\}]^{1.5} + [\max\{0,\ d(i,j) - 1500\}]^{2} .$$

4. COMPUTATIONAL TESTING

Our approach has been tested on two real problems in Canada, we will refer to these problems as the Postal station X and the Postal station Y.

4.1 The Postal station X

This postal network is composed of 353 nodes and 1,116 arcs (see Figure 12), which represents a total service time of 172 hours and 4 minutes. Table 1 summarizes the results obtained.

solution	number of letter carrier	average time	standard deviation	total transport time	average transport time
actual solution	25* (+1)	480,81 99,63	9,31 --	942,5 4,8	37,7 4,8
our solution	25	481,45	8,31	906,5	36,26

Table 1: Results on Postal station X

Note: One letter carrier works on a part-time basis on a route of 99.63 minutes, with a transportation time of 4.84 minutes. This route is not included in the statistics given here.

In the Postal station X, we succeeded in eliminating a part time route of 99.63 minutes mostly by reducing the total transportation time (4,3% reduction) and by slightly increasing the total work time.

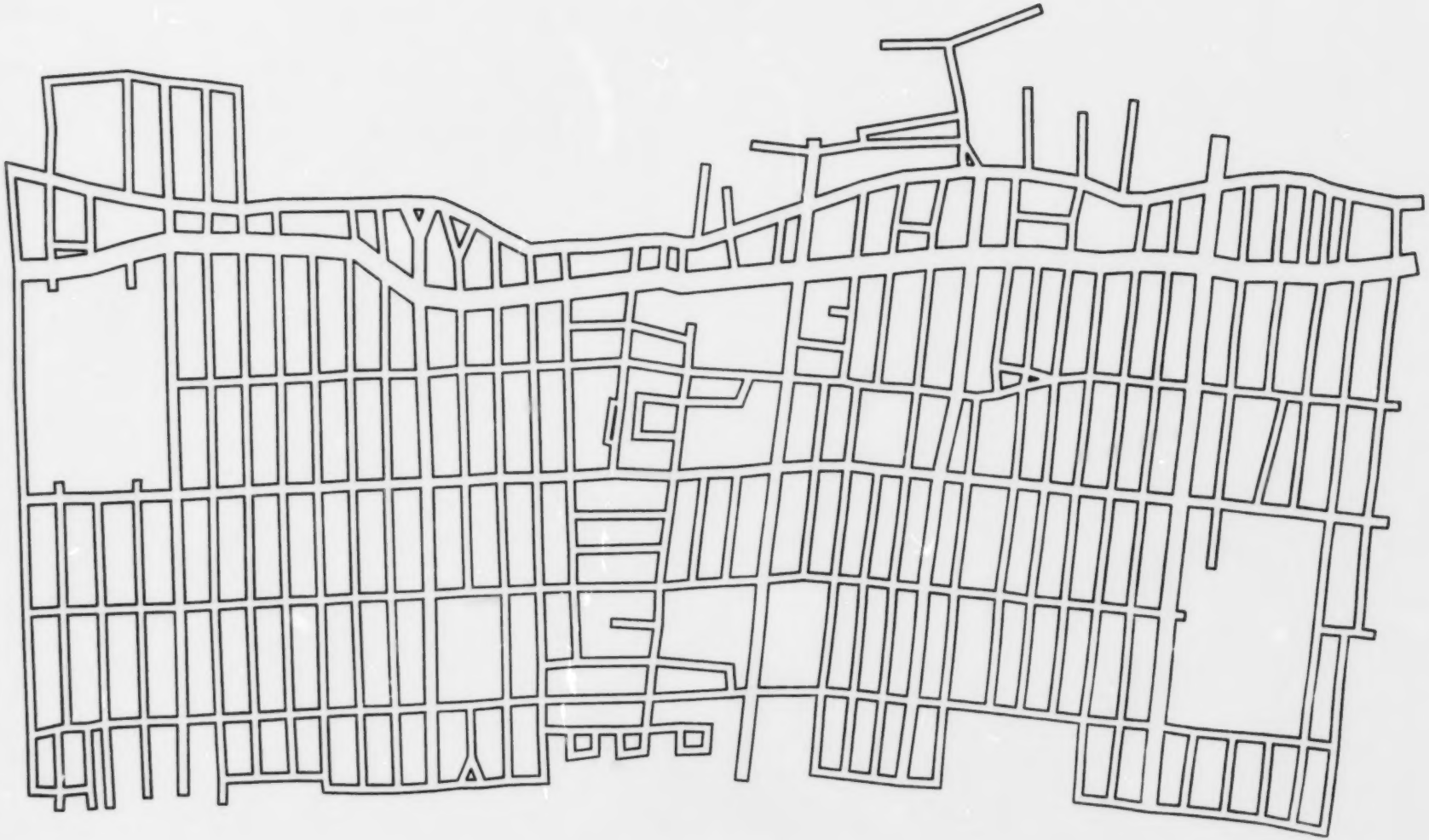


Figure 12: BOURASSA Postal Station
353 nodes and 1,116 arcs

Figure 13: BEAUPORT Postal Station
683 nodes and 1,972 arcs

4.2 The Postal station Y

This postal network is composed of 683 nodes and 1,972 arcs (see Figure 13), which represents a total service time of 170 hours and 38 minutes. Table 2 summarizes the results.

solution	number of letter carrier	average time	standard deviation	total transport time	average transport time
actual solution	21	467,9	10,36	663,29	31,59
our solution	20	471,18	10,07	340,86	17,04

Table 2: Results on Postal station Y

In the Postal station Y, it was possible to reduce by one the number of postmen mainly by reducing the total travel time by 48%. Work time was slightly increased but the number of workdays exceeding the 480 minute capacity was approximately the same.

5. CONCLUSION

The method we have developed for Canada Post Corporation has several advantages. First, we have demonstrated that savings can be achieved and be very significant if generalized over all CPC operations. Secondly, the method is very flexible and several different strategies can be used if required depending on the geographical structure of the network, the existence or not of a transit network, the postal station location, etc. By experimentation, several sets of selection functions could be prepared to generate good solutions in different specific cases. Finally, because of its modularity, it is easy to let the user interact with the general algorithm at various steps in order to imput his/her judgement and experience. What remains to be done and what is essential for a successful

implementation is to embed this procedure in an efficient user friendly and interactive-graphics system to stimulate the interaction between the user and the system. We also believe that the general heuristic we have proposed here could easily be adapted for most of the other CARP problems.

ACKNOWLEDGEMENTS

We gratefully acknowledge discussions with Michel Gendreau on the model (CPC), and Yves Simard, Eric Landry and Lofs Feinholz of the Canada Post Corporation on the description of the problem.

REFERENCES

1. ASSAD, A., PEARN, W.L., and GOLDEN, B. (1985) "The Capacitated Chinese Postman Problem: Lower Bounds and Solvable Cases", working paper #85032, University of Maryland.
2. BODIN, L. and GOLDEN, B. (1981) "Classification in Vehicle Routing and Scheduling", Networks 11(2), 97-108.
3. CHRISTOFIDES, N. (1973) "The Optimum Traversal of a Graph" OMEGA 1(6), 719-732.
4. DESROCHERS, M., LENSTRA, J.K., and SAVELSBERGH, M.W.P. (1987) "A Classification Scheme for Vehicle Routing and Scheduling Problems", Report OS-R87XX, Centre for Mathematics and Computer Sciences, Amsterdam, The Netherlands.
5. EDMONDS, J. and JOHNSON, E. (1973) "Matching, Euler Tours and the Chinese Postman" Mathematical Programming 4, 88-124.
6. EULER, L. (1953) "The Konigsberg Bridges", Scientific American 189, 66-70.
7. GOLDEN, B. et WONG, R. (1981) "Capacitated Arc Routing Problem", Networks, vol. 11, 305-315.
8. GOLDEN, B., DEARMON, J.S., and BAKER, E.K. (1985) "Algorithms for the Capacitated Chinese Postman Problem", working paper MS/S #81-024, University of Maryland.

9. LAPORTE, G. (1988) "Location Routing Problems", in: Vehicle Routing: Methods and Studies, Golden B.L. and Assad A.A. (eds.) North Holland, 163-198.

10. LEVY, L., BODIN, L. (1988) "Scheduling in the Postal Carriers for the United States Postal Service - An Application of Arc Partitioning and Routing", in: Vehicle Routing: Methods and Studies, Golden B.L. and Assad A.A. (eds.) North Holland, 359-395.

11. LEVY, L., BODIN, L. (1989) The Arc Location Routing Problem (this issue).

12. LEVY, L. (1987) "The Walking Line of Travel Problem: An Application of Arc Routing and Partitioning", Ph.D. dissertation, University of Maryland at College Park.

13. MINIEKA, E. (1979) "The Chinese Postman Problem for Mixed Networks", Management Science 25(7), 643-648.

14. PAPADIMITRIOU, C. (1976) "On the Complexity of Edge Traversing", J. Assoc. Comput. Math., vol. 23.

15. PEARN, W.-L. (1984) "The Capacitated Chinese Postman Problem", Ph.D. dissertation, University of Maryland at College Park.

16. PEARN, W., ASSAD, A. and GOLDEN, B. (1986) "Transforming Arc Routing into Node Routing Problems", working paper MS/86-006, University of Maryland.

17. POTVIN, J.-Y., ROUSSEAU, J.-M. and LAPALME, G. (1987) "ALTO: A Computer System for the Design and Experimentation of Routing Algorithms", publication #525, Centre de recherche sur les transports, Université de Montréal.

18. STRICKER, R. (1970) "Public Sector Vehicle Routing the Chinese Postman Problem", Master's theses at M.I.T.